Figure 1 - Verilog-AMS Circuit Example - Prior Art

```
// divider from Verilog-ams standarization committee public circuits
`timescale 10ns/1ns
`include "disciplines.h"
`include "connect.h"

module top;
  reg clk;
  wire sys_clk;

  // digital constructs
  initial clk = 0;
  always #5 clk = ~clk;            // 1uS Clock Generator
  assign sys_clk = clk;

  // instantiations
  zdetect my_dev(sys_clk, divout);
  lpf #(.tau(1.59e-8)) tenMlpf(divout, tenMout);
endmodule

module zdetect(in,out);
  input in;
  output out;
  electrical in,out;
  integer n, state;
  parameter div = 5;

  // analog blocks code analog circuits as equations
analog begin
  @(cross(V(in) - 2.5, +1)) n = n + 1;
  if (n >= div )begin
    if (state == 0) state = 1;
    else state = 0;
    n = 0;
  end
  V(out) <+ state * 5;
end endmodule

module lpf(in, out);
  inout in, out;
  electrical in, out;
  parameter real tau = 1e-3;

analog
  begin
   V(out) <+ laplace_nd(V(in), {1.0}, {1.0, tau});
  end
endmodule
```

Fig. 2

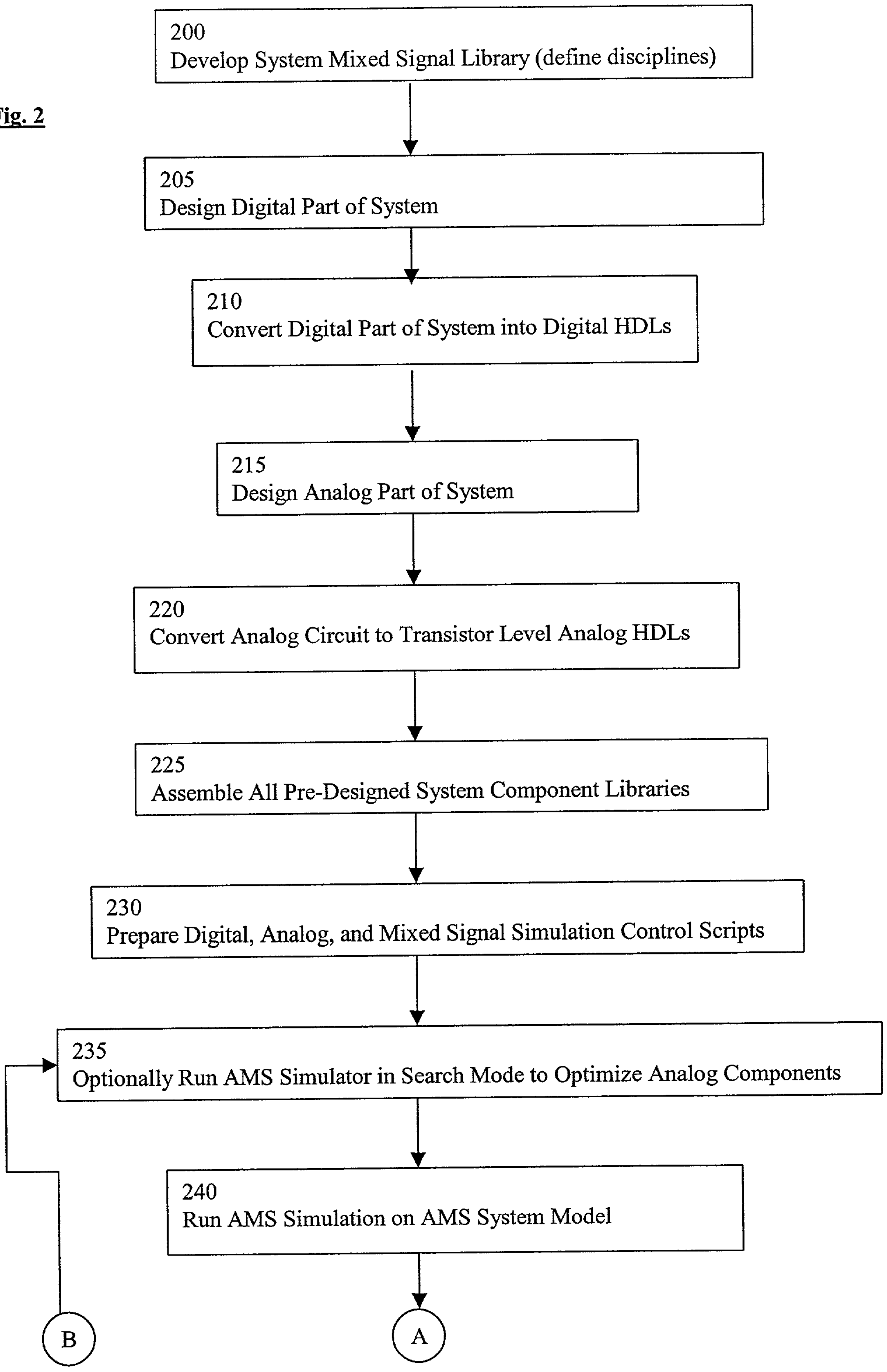
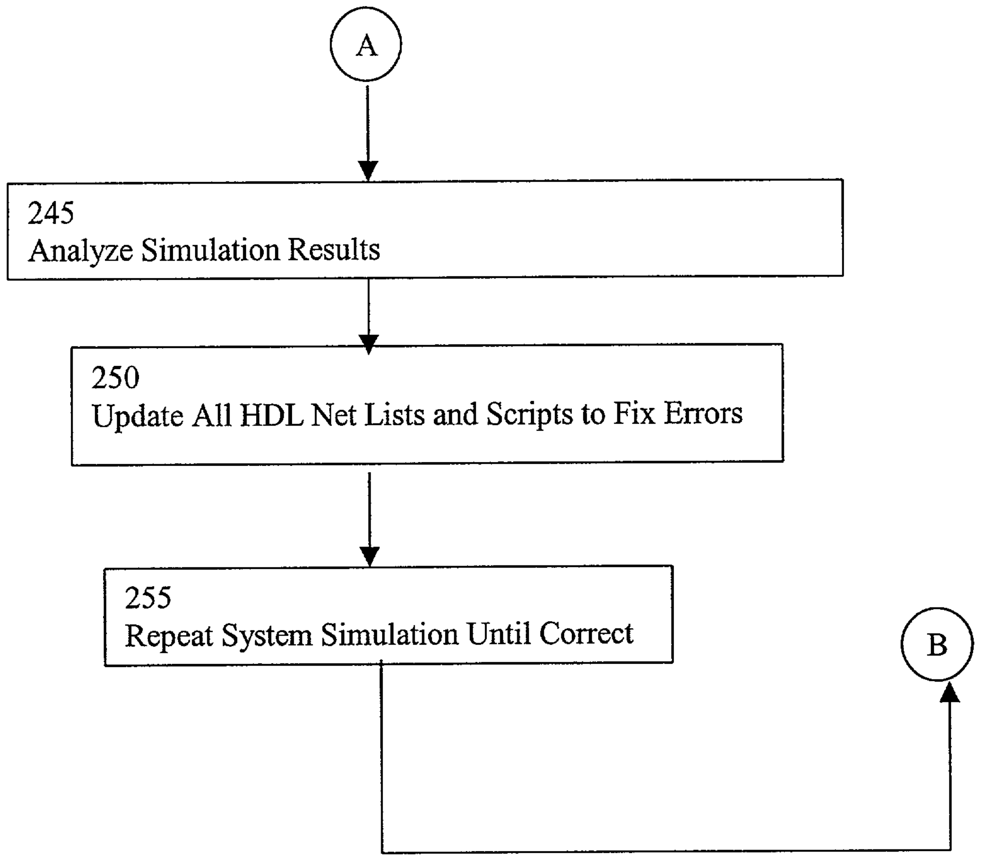
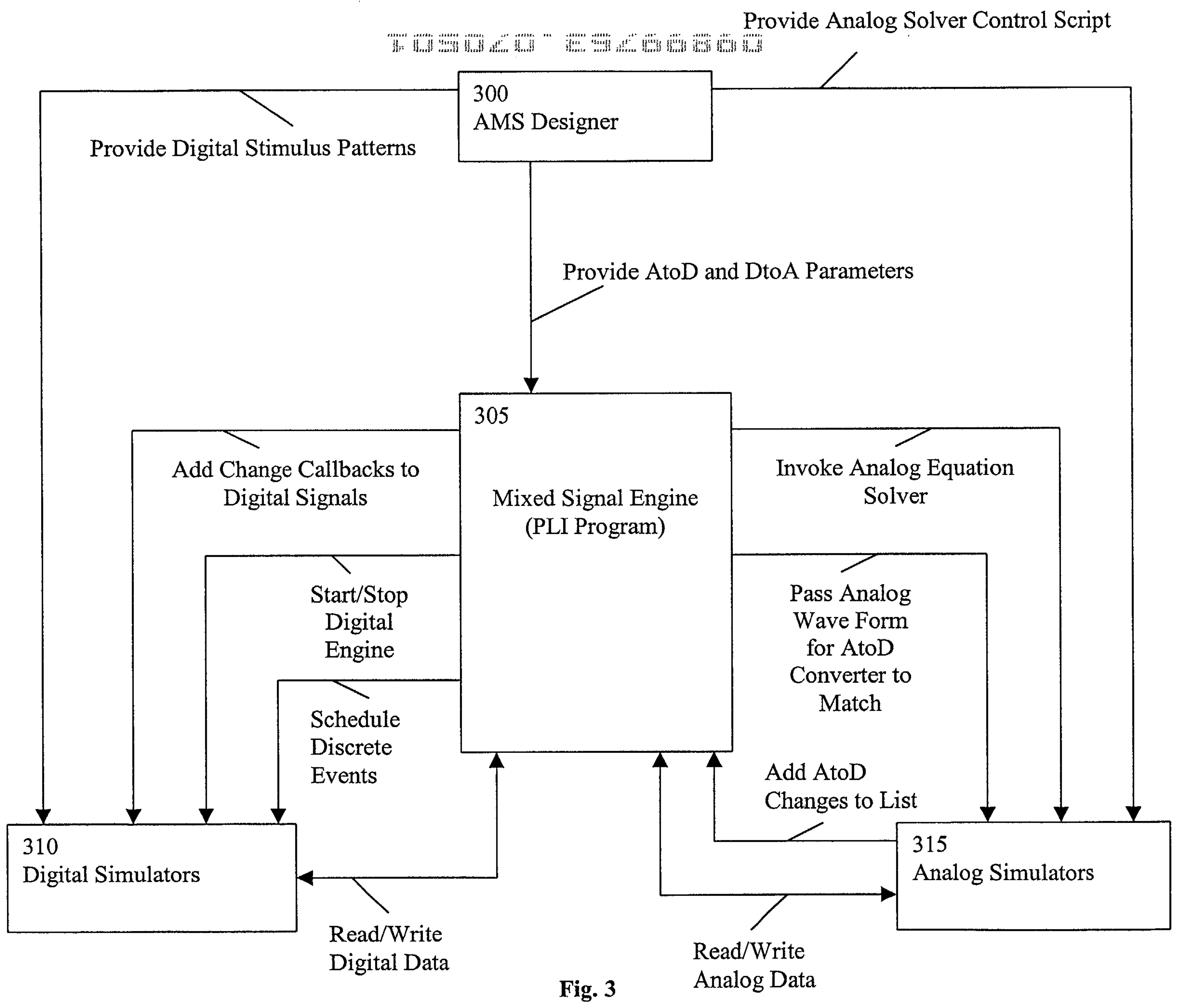

200
Develop System Mixed Signal Library (define disciplines)
205
Design Digital Part of System
210
Convert Digital Part of System into Digital HDLs
215
Design Analog Part of System
220
Convert Analog Circuit to Transistor Level Analog HDLs
225
Assemble All Pre-Designed System Component Libraries
230
Prepare Digital, Analog, and Mixed Signal Simulation Control Scripts
235
Optionally Run AMS Simulator in Search Mode to Optimize Analog Components
240
Run AMS Simulation on AMS System Model
B
A

Fig. 2 (cont'd)


A
245
Analyze Simulation Results
250
Update All HDL Net Lists and Scripts to Fix Errors
255
Repeat System Simulation Until Correct
B



Provide Analog Solver Control Script
300
AMS Designer
Provide Digital Stimulus Patterns
Provide AtoD and DtoA Parameters
305
Mixed Signal Engine
(PLI Program)
Add Change Callbacks to Digital Signals
Start/Stop Digital Engine
Schedule Discrete Events
Invoke Analog Equation Solver
Pass Analog Wave Form for AtoD Converter to Match
Add AtoD Changes to List
310
Digital Simulators
315
Analog Simulators
Read/Write Digital Data
Read/Write Analog Data

Fig. 3

Fig. 4

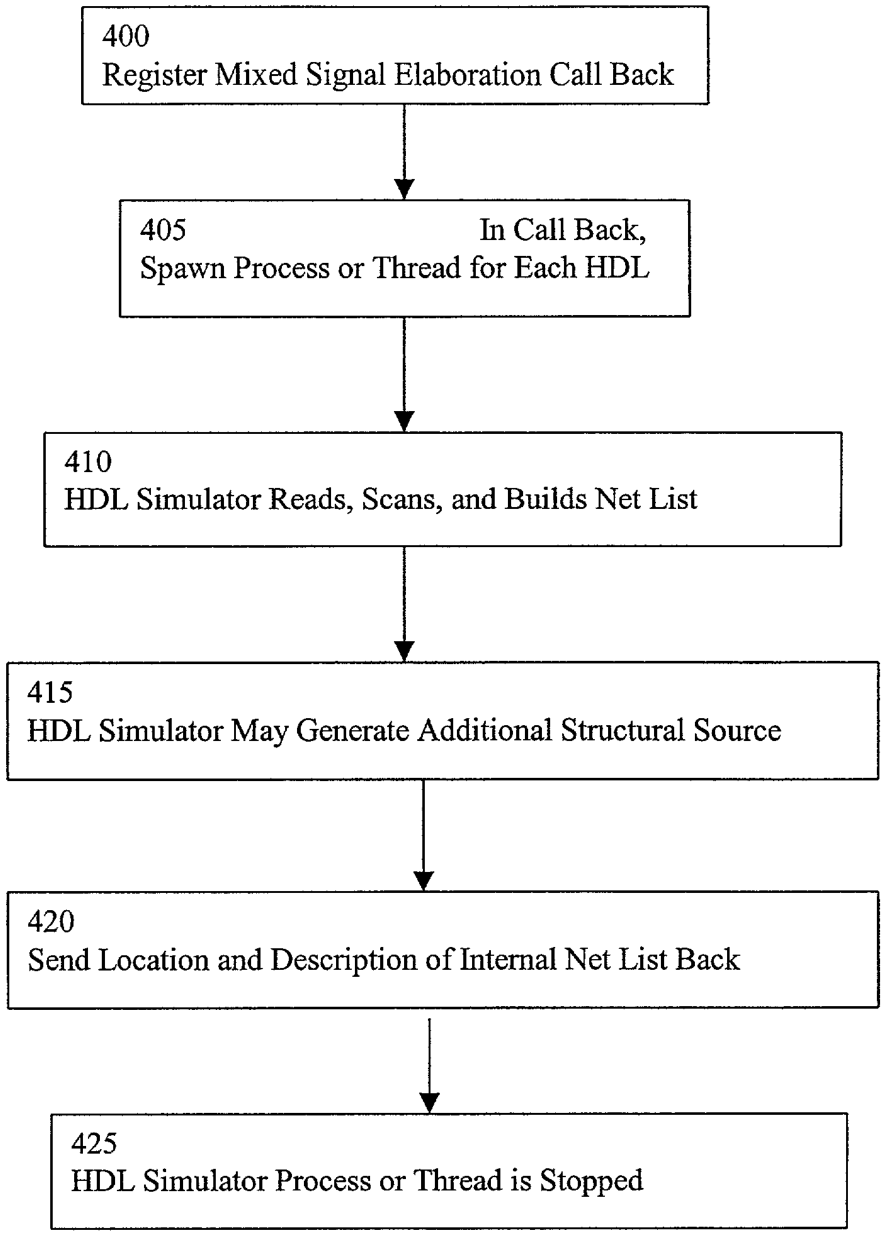

400
Register Mixed Signal Elaboration Call Back
405 In Call Back,
Spawn Process or Thread for Each HDL
410
HDL Simulator Reads, Scans, and Builds Net List
415
HDL Simulator May Generate Additional Structural Source
420
Send Location and Description of Internal Net List Back
425
HDL Simulator Process or Thread is Stopped

Fig. 5

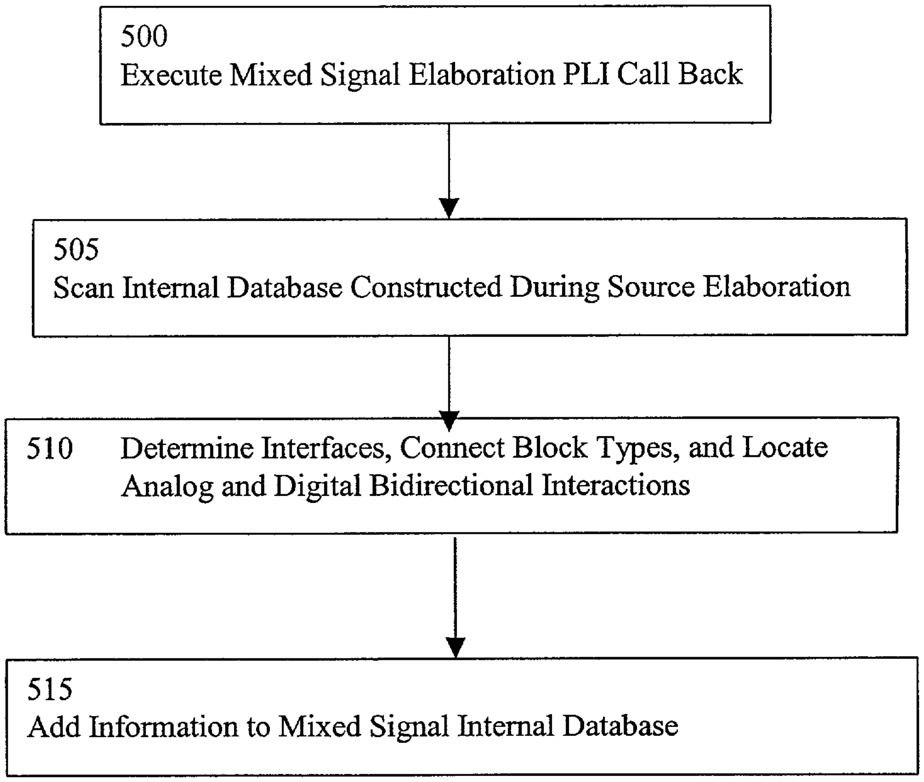
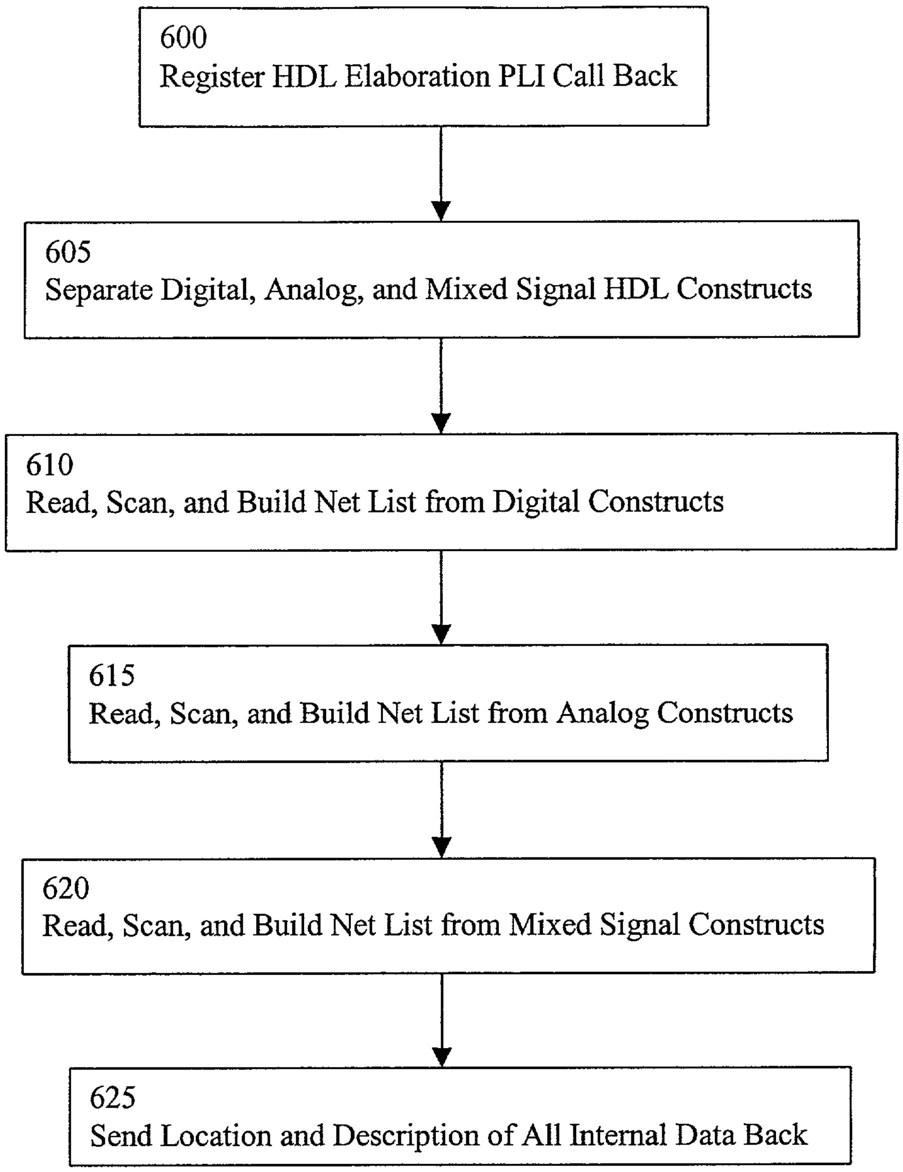

500
Execute Mixed Signal Elaboration PLI Call Back
505
Scan Internal Database Constructed During Source Elaboration
510 Determine Interfaces, Connect Block Types, and Locate Analog and Digital Bidirectional Interactions
515
Add Information to Mixed Signal Internal Database

Fig. 6



600
Register HDL Elaboration PLI Call Back
605
Separate Digital, Analog, and Mixed Signal HDL Constructs
610
Read, Scan, and Build Net List from Digital Constructs
615
Read, Scan, and Build Net List from Analog Constructs
620
Read, Scan, and Build Net List from Mixed Signal Constructs
625
Send Location and Description of All Internal Data Back

Fig. 7

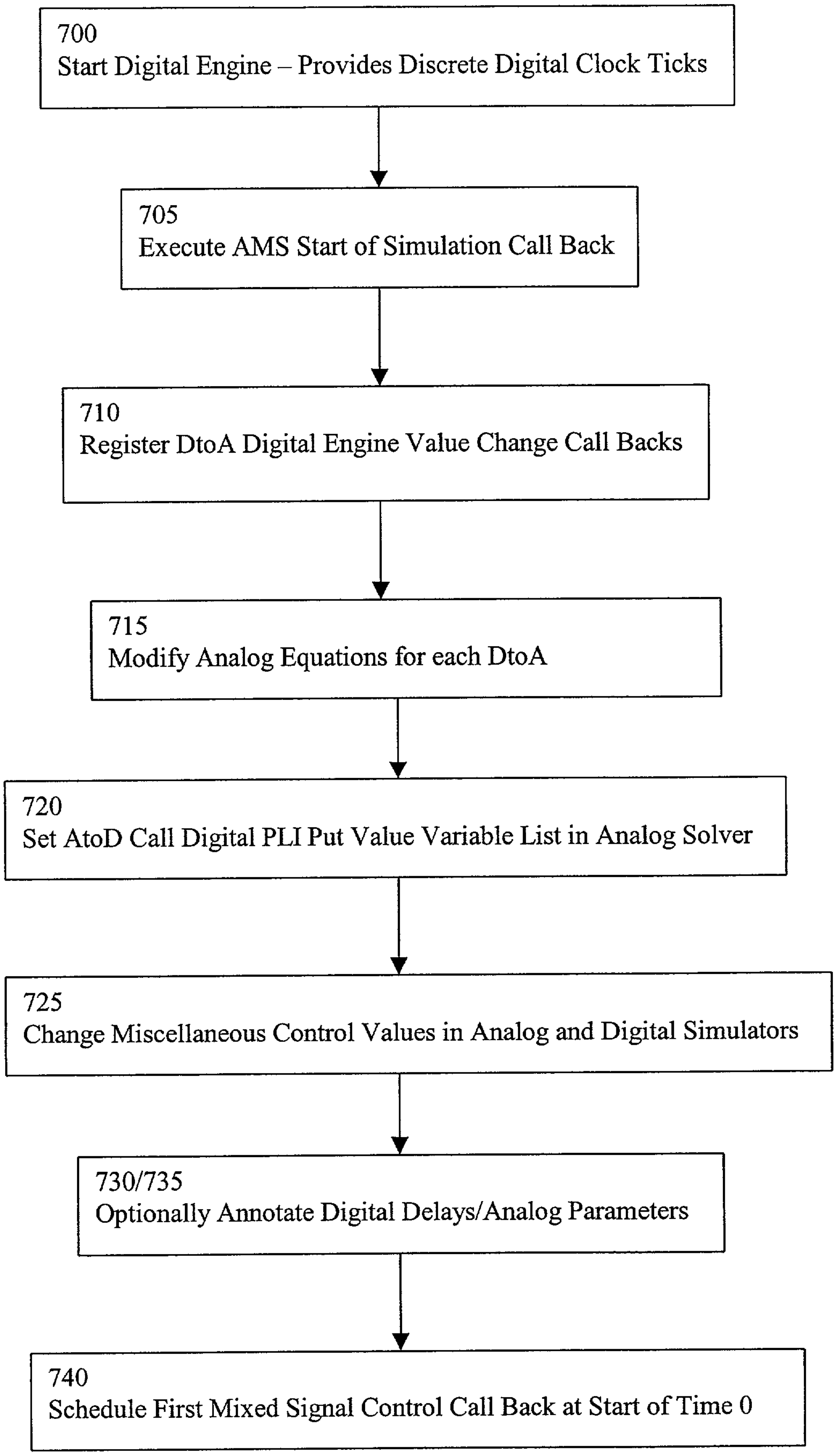

700
Start Digital Engine – Provides Discrete Digital Clock Ticks
705
Execute AMS Start of Simulation Call Back
710
Register DtoA Digital Engine Value Change Call Backs
715
Modify Analog Equations for each DtoA
720
Set AtoD Call Digital PLI Put Value Variable List in Analog Solver
725
Change Miscellaneous Control Values in Analog and Digital Simulators
730/735
Optionally Annotate Digital Delays/Analog Parameters
740
Schedule First Mixed Signal Control Call Back at Start of Time 0

Fig. 8

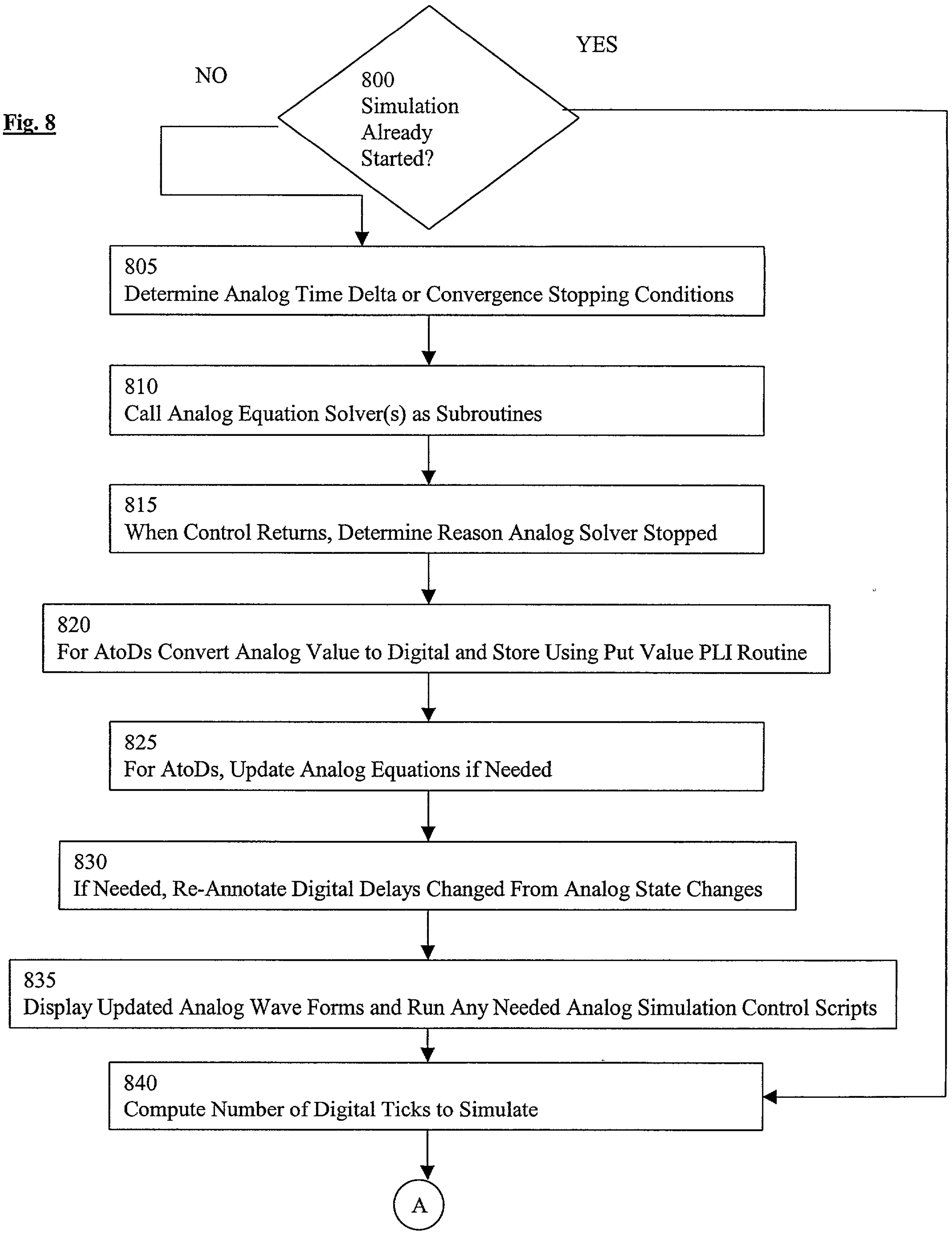
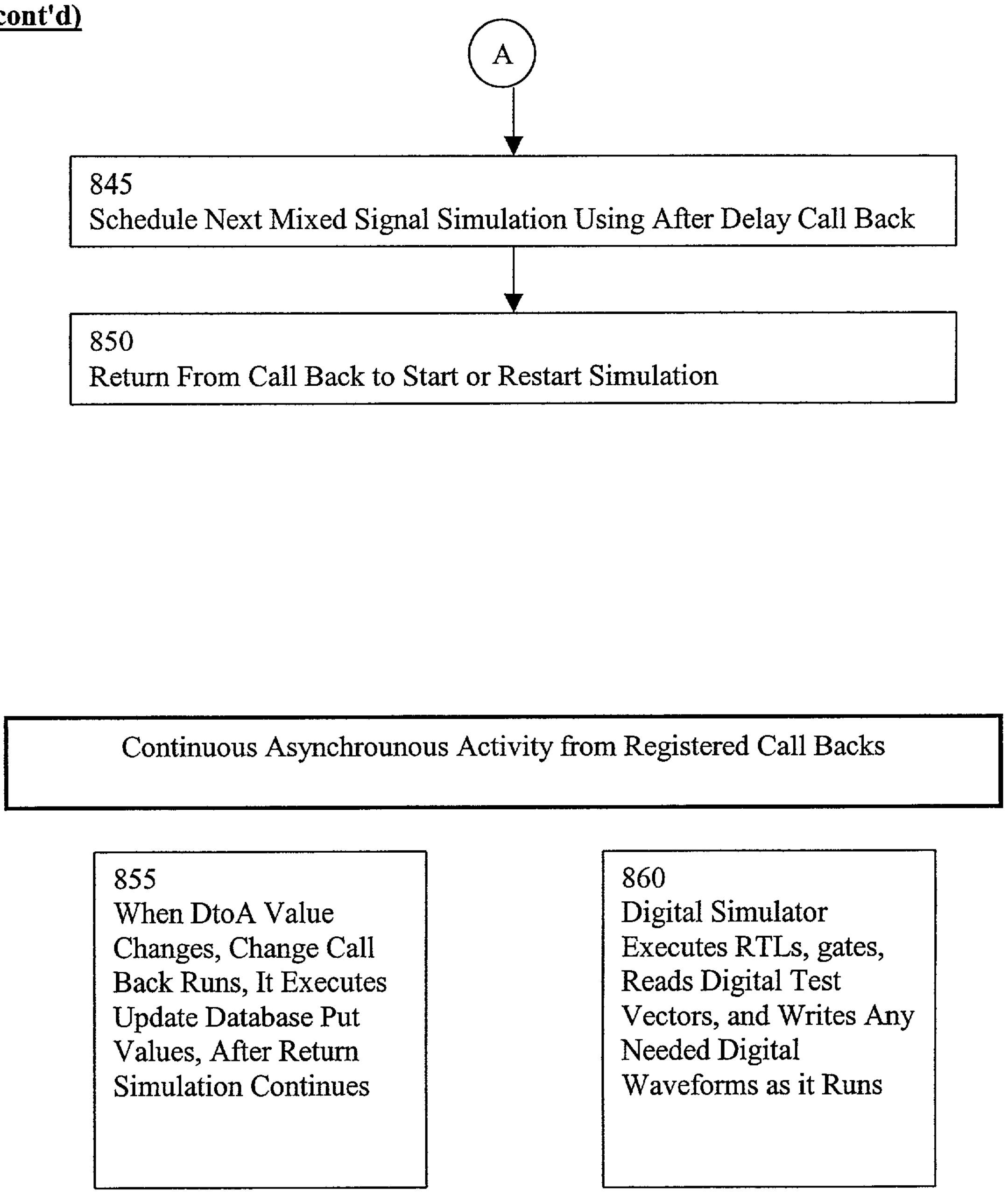

800
Simulation Already Started?
NO
YES
805
Determine Analog Time Delta or Convergence Stopping Conditions
810
Call Analog Equation Solver(s) as Subroutines
815
When Control Returns, Determine Reason Analog Solver Stopped
820
For AtoDs Convert Analog Value to Digital and Store Using Put Value PLI Routine
825
For AtoDs, Update Analog Equations if Needed
830
If Needed, Re-Annotate Digital Delays Changed From Analog State Changes
835
Display Updated Analog Wave Forms and Run Any Needed Analog Simulation Control Scripts
840
Compute Number of Digital Ticks to Simulate
A

Fig. 8 (cont'd)


A
845
Schedule Next Mixed Signal Simulation Using After Delay Call Back
850
Return From Call Back to Start or Restart Simulation
Continuous Asynchrounous Activity from Registered Call Backs
855
When DtoA Value Changes, Change Call Back Runs, It Executes Update Database Put Values, After Return Simulation Continues
860
Digital Simulator Executes RTLs, gates, Reads Digital Test Vectors, and Writes Any Needed Digital Waveforms as it Runs

9.2.4 The synchronization loop

The digital and analog kernels shall be synchronized so neither computes results which the other is ineligible to accept. The synchronization algorithm can exploit characteristics of the analog and digital kernels described in the next section. A sample run is shown in Figure 9-4.

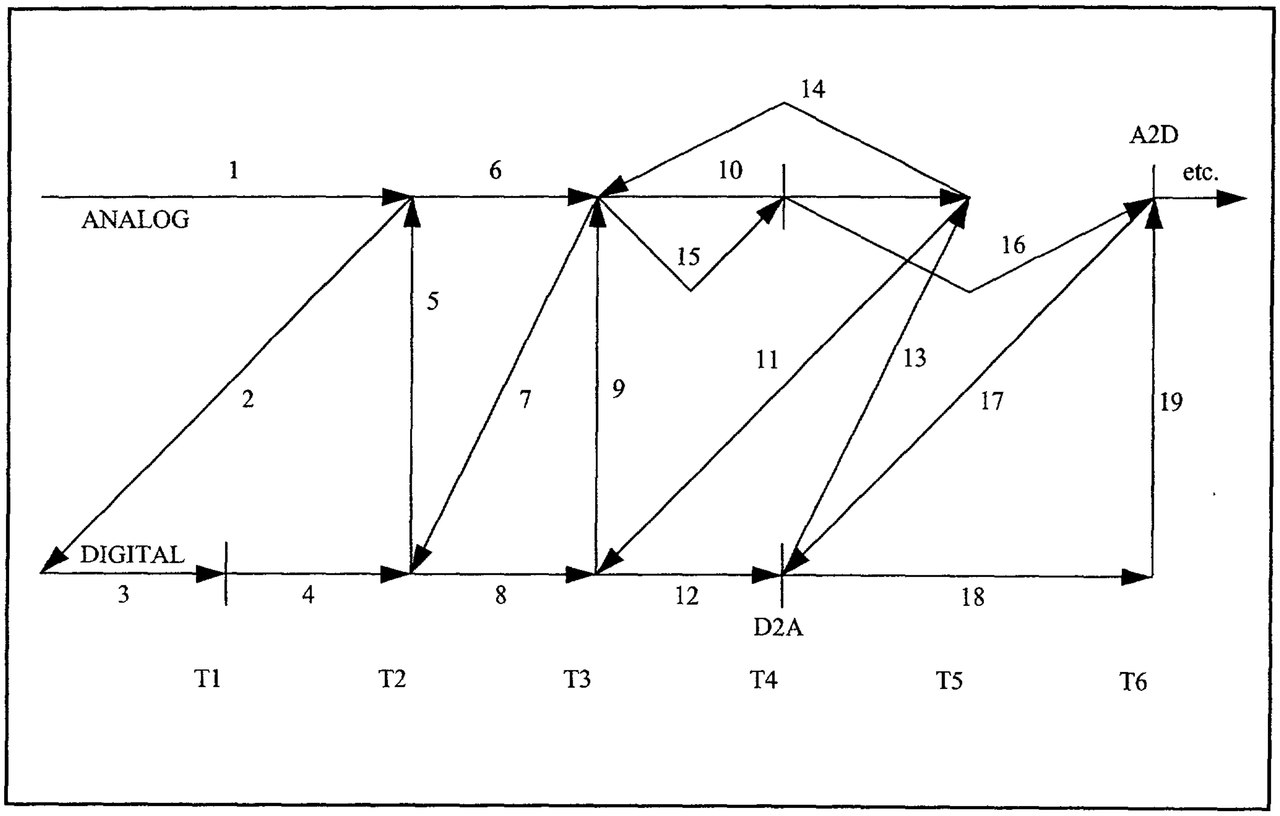


Figure 9-4 Sample run

1. The Analog engine begins transient analysis and sends state information to the Digital engine (1,2).
2. The Digital engine begins to run using its own time steps (3); however, if there is no D2A event, the Analog engine is not notified and the digital engine continues to simulate to until it can not advance its time without surpassing the time of the analog solution (4). Control of the simulation is then returned to the analog engine (5). This process is repeated (7,8,9,10, and 11).
3. If the Digital engine produces a D2A event (12), control of the simulation is returned to the Analog engine (13). The analog engine returns to the point at which the digital engine last surrendered control (14). The Analog engine recalculates the analog solution up to the time when the D2A event occurred (15). The Analog engine then takes the next time step (16).

FIG. 9 (1 OF 3)

4. If the Analog engine produces an A2D event, it returns control to the Digital engine (17), which simulates up to the time of the A2D event and then surrenders control (18 and 19).

5. This process continues until transient analysis is complete.

9.2.5 Assumptions about the analog and digital algorithms

1. Advance of time in a digital algorithm

- The digital simulation has some minimum time granularity and all digital events occur at a time which is some integer multiple of that granularity.
- The digital simulator can always accept events for a given simulation time provided it has not yet executed events for a later time. Once it executes events for a given time, it can not accept events for an earlier time.
- The digital simulator can always report the time of the most recently executed event and the time of the next pending event.

2. Advance of time in an analog algorithm

- The analog simulator advances time by calculating a sequence of solutions. Each solution has an associated time which, unlike the digital time, is not constrained to a particular minimum granularity.
- The analog simulator can not tell for certain the time when the next solution converges. Thus, it can tell the time of the most recently calculated solution, but not the time of the next solution.
- In general, the analog solution is a function of one or more previous solutions. Having calculated the solution for a given time, the analog simulator can either accept or reject that solution; it can not calculate a solution for a future time until it has accepted the solution for the current time.

3. Analog to digital events

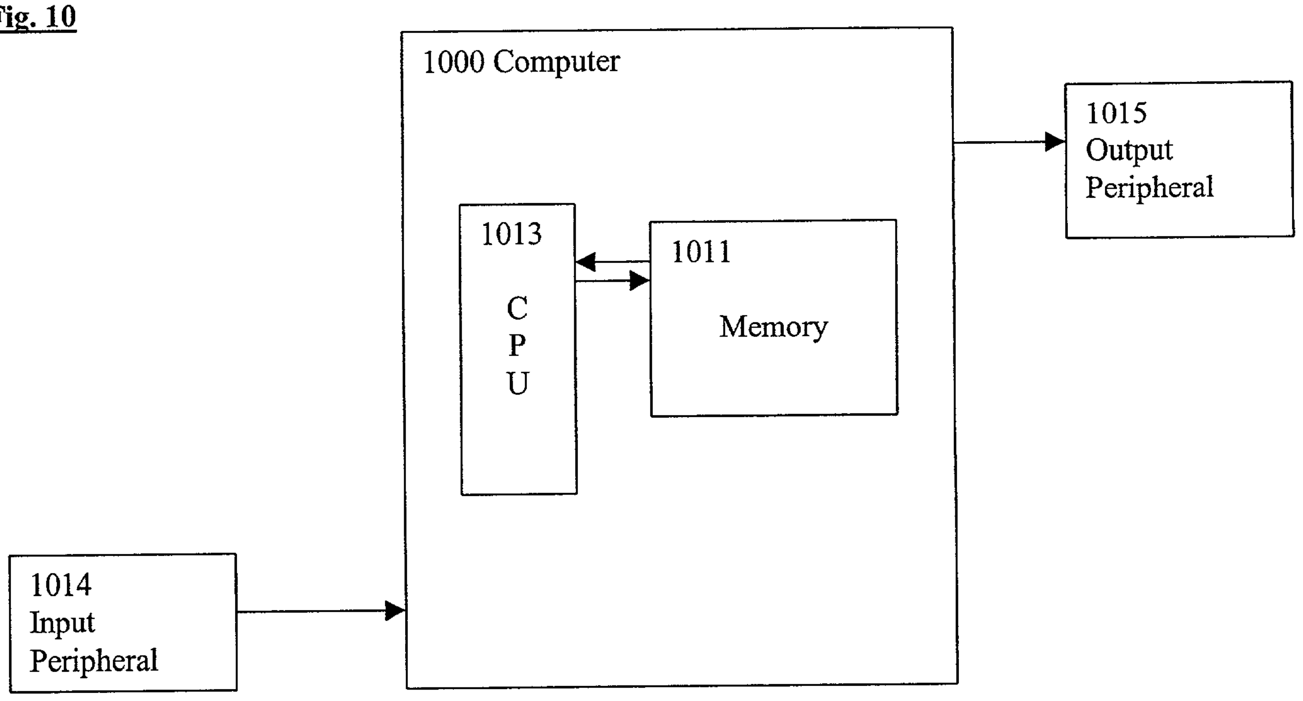
- Analog to digital events are generated by conversion elements (which are analog/digital behavioral models) when evaluated by the analog simulator.
- Analog events (e.g., `cross`, `initial_step`, and `final_step`) cause an analog solution of the time where they occur.
- Thus, any analog to digital event is generated as the result of a particular transient solution. This means events can stay associated with the solution which produced them until they are passed to the digital simulator, then they can be rejected along with the solution if it is rejected.

FIG. 9 (2 OF 3)

4. Digital to analog events shall cause an analog solution of the time where they occur.

FIG. 9 (3 OF 3)

Fig. 10


1000 Computer
1013
C
P
U
1011
Memory
1015
Output
Peripheral
1014
Input
Peripheral